**Day 1: Introduction to Python**

**A Brief History**

* **Creator:** **Guido van Rossum**.
* **Development Period:** The initial development began in **1989**.
* **First Public Release:** Python was officially released in **1991** with version 0.9.0.
* **Python's Logo:** The iconic two-snakes logo was designed by Guido's brother, **Justin van Rossum**.
* **Core Philosophy:** Python was designed with a strong emphasis on **code readability** and **simplicity**. The goal was to create a language that is easy to learn and write, allowing developers to build applications faster. It's considered a high-level, general-purpose language, meaning it can be used for a wide variety of tasks (web development, data analysis, AI, etc.).

**Key Questions & Concepts**

* **Q1: What is the difference between a compiler and an interpreter?**
  + **Compiler:** A compiler takes your entire program (source code) and translates it into machine code (a language the computer's processor can understand) all at once. This creates a standalone executable file (like a .exe on Windows). You run this file to execute the program. Languages like C, C++, and Java are primarily compiled.
  + **Interpreter:** An interpreter reads your source code line by line, translates each line into machine code, and executes it immediately. It doesn't create a separate executable file. **Python is an interpreted language**. This is why you can type a command and see the result instantly, which makes it great for learning and testing ideas quickly.
* **Q2: What is Modula-3?**
  + Modula-3 is another programming language that was a major influence on Guido van Rossum when he was creating Python. He liked some of its features, such as its syntax and its approach to handling errors (exceptions), and incorporated similar ideas into Python.
* **Q3: Python's open-source certification.**
  + Python is an **open-source** language, which means its source code is freely available for anyone to use, modify, and distribute. It's managed by the **Python Software Foundation (PSF)**, a non-profit organization. The PSF holds the intellectual property rights to Python and manages its open-source license. This license ensures Python remains free and open for everyone. There isn't a specific "open-source certification" that Python "got," but rather its entire existence is based on this open and collaborative model.
* **Q4: Why was Java more famous than Python initially?**
  + This is a great question! While Python was developed first, Java (released in 1995) gained massive popularity in the late 90s and 2000s for a few key reasons:
    - **Corporate Backing:** Java was created and heavily marketed by Sun Microsystems, a large tech company.
    - **The Web:** Java was famous for "applets," small applications that could run inside web browsers, which was a huge deal at the time.
    - **Enterprise Focus:** It was designed for large-scale, enterprise-level applications, making it very popular in the corporate world.
    - **Familiar Syntax:** Its syntax is similar to C++, which was the dominant language at the time, making it easier for existing programmers to adopt.

**Understanding Data in Python**

Before we dive into code, let's clarify the most fundamental concepts.

**What is Data?**

As your notes correctly state, **data is any observation that is recorded**. In programming, this "observation" can be a number, a piece of text, a true/false value, or a more complex collection of these things.

**What are Data Types?**

A **data type** is a classification that tells the computer how to interpret a piece of data. It defines what kind of values a variable can hold and what kind of operations can be performed on it.

* **Analogy:** Think of data types like different kinds of containers. A water bottle is designed to hold liquids (like floating-point numbers), while a shoebox is designed to hold shoes (like a string of text). You use the right container for the right item.

**Mutable vs. Immutable Data Types**

This is a crucial concept in Python.

* **Immutable (Unchangeable):** The value of an immutable object **cannot be changed** after it is created. If you want to "change" it, Python actually creates a *new* object in memory. The data types we will cover in these notes (**int, float, complex, string, tuple**) are all immutable.
* **Mutable (Changeable):** The value of a mutable object **can be changed** after it is created. Examples include lists, dictionaries, and sets (which you'll likely learn about later).

**Core Python Data Types**

Here we'll explore the fundamental data types you practiced with, along with your code examples.

**1. Numerical Types**

Numerical data types are used to store and process numerical values. They are immutable, meaning the value of their object cannot be changed once created. Python supports three distinct numerical types.

**Integer (int)**

* **Definition:** An integer is a whole number, which can be positive, negative, or zero, without any decimal or fractional part. In Python 3, integers have arbitrary precision, meaning they can be as large or small as your computer's memory allows.
* **Use Case:** Used for counting discrete objects, indexing items in a sequence, and any mathematical calculation that does not require fractional precision.
* **Code Example**

# int

print(123)

print(-456)

print(type(123))

* **Explanation**
  + print(123): This line outputs the integer value 123.
  + print(-456): This demonstrates that integers can also be negative.
  + print(type(123)): The built-in type() function inspects the object 123 and returns its data type. The output <class 'int'> confirms that whole numbers are classified as integers in Python.

**Float (float)**

* **Definition:** A float, or floating-point number, is a number that contains a decimal point. It is used to represent real numbers and is essential for calculations that require fractional precision.
* **Use Case:** Ideal for scientific calculations, measurements (like height or weight), and financial applications where decimal values are necessary.
* **Code Example**

Python

# float

print(1.10)

print(-2.25)

print(type(-2.25))

* **Explanation**
  + print(1.10): This displays the floating-point number. Python may display it as 1.1, automatically removing non-significant trailing zeros, but the value is stored with full precision.
  + print(-2.25): This shows a negative floating-point number.
  + print(type(-2.25)): The type() function confirms that any number written with a decimal point is recognized as a <class 'float'>.

**Complex (complex)**

* **Definition:** A complex number is a number with both a real and an imaginary part. It is expressed in the form a+bj, where 'a' is the real part, 'b' is the imaginary part, and j represents the imaginary unit (−1​).
* **Use Case:** Commonly used in advanced mathematics, physics, signal processing, and electrical engineering.
* **Code Example**

Python

# complex

print(3+4j)

print(type(3+4j))

* **Explanation**
  + print(3+4j): This prints the complex number where 3 is the real component and 4 is the imaginary component.
  + print(type(3+4j)): This confirms that Python has native support for complex numbers, classifying this value as <class 'complex'>.

**2. String (str)**

* **Definition:** A string is an **ordered, immutable sequence of characters**. It is used to represent textual data. "Immutable" means that once a string is created, it cannot be modified.
* **Creation:**
  + **Single (') and Double (") Quotes:** Functionally identical and used for single-line strings. Their primary difference is convenience; use one type of quote to easily enclose a string that contains the other.
    - Example: print("It's a beautiful day.")
  + **Triple Quotes (''' or """)**: Used for multi-line strings and docstrings. They preserve all whitespace and line breaks exactly as they are typed.
* print('''This text

spans multiple

lines.''')

**3. Tuple (tuple)**

* **Definition:** A tuple is an **ordered, immutable collection of items**. "Ordered" means the items are stored in a specific sequence that will not change. "Immutable" means that after a tuple is created, you cannot add, remove, or change its elements. Tuples are created by enclosing items in parentheses ().
* **Key Characteristics:**
  + **Heterogeneous:** A tuple can contain a mix of different data types (e.g., an integer, a string, and a float).
  + **Indexed:** You can access items in a tuple by their numerical index, starting from 0.
  + **Efficient:** Tuples are generally more memory-efficient and faster to process than lists (their mutable counterpart).
* **Code Example**

# A tuple with mixed data types

user\_data = (101, "Alice", 25, "Canada")

print(user\_data)

print(type(user\_data))

# Another example

print(("I have", 1, 'apple'))

* **Explanation**
  + print(user\_data): This line prints the tuple containing an integer, a string, another integer, and another string. This is a heterogeneous collection.
  + print(type(user\_data)): This confirms that the data structure created with parentheses is a <class 'tuple'>.
  + print(("I have", 1, 'apple')): This is another example of a heterogeneous tuple being created and printed directly. It effectively groups related but different types of data into a single, unchangeable unit.

**Useful Built-in Functions**

Your notes used a few handy built-in functions. Let's look at one more you experimented with.

**The chr() function**

The chr() function does the opposite of what ord() does. It takes an integer (a Unicode code point) and returns the character that corresponds to it.

**Code Examples**

print(chr(1000464))

chr(2323)

**Explanation**

1. print(chr(1000464)): You provided the integer 1000464 to chr(). This number corresponds to a specific character in the vast Unicode table, which is then printed. The output is '󴐐'.
2. chr(2323): The number 2323 corresponds to the Devanagari character 'ओ' in Unicode.

**Quick Revision Summary**

* **Python:** A high-level, interpreted programming language created by Guido van Rossum, released in 1991. It prioritizes code readability.
* **Interpreter:** Executes code line by line (this is what Python does).
* **Compiler:** Translates the entire program at once before execution.
* **Data Type:** A classification of data that tells the program how to use it (e.g., as a number, text, etc.).
* **Immutable:** The value cannot be changed after creation.
* **Core Data Types:**
  + **int:** Whole numbers (e.g., 10, -50).
  + **float:** Numbers with decimals (e.g., 99.9, -0.123).
  + **complex:** Numbers with a real and imaginary part (e.g., 5+2j).
  + **str:** Text, created with ' ', " ", or ''' '''.
  + **tuple:** An ordered, immutable collection of items in ( ).
* **Key Functions:**
  + **print():** Displays output to the console.
  + **type():** Returns the data type of a value.
  + **chr():** Returns a character from a Unicode integer code.